École Polytechnique de Louvain

[LSINF 1225]
Conception Orientée Objet
et Gestion de Données

Groupe F — Année 2014–2015

Travail 1

La Modélisation de Données

Auteurs :

Mathieu Delandmeter	6240 – 13 – 00
Nathan Gillain	7879 – 12 – 00
Maxime Hanot	6591 – 13 – 00
Alexandre Jadin	4844 – 13 – 00
Thomas Marissal	8217 – 13 – 00
Edouard Vangangel	2243 – 09 – 00

Professeur :
Kim Mens

Tuteur :
Benoît Baufays

3 mars 2015

Introduction

Dans le cadre du cours de conception orientée objet et gestion de données, dispensé par le Professeur Kim Mens, il nous a été demandé d'implémenter une application *Android* de type *"Gestion de bar"*. Dans un premier temps, nous avons dû élaborer une base de donnée avec l'outil SQLite, tâche qui a nécessité plusieurs étapes. Dès lors, ce document, premier rapport de ce projet, a pour but d'expliquer la démarche que nous avons suivie ainsi que les délivrables que nous avons créés.

Démarche

Tout d'abord, nous avons dû déterminer les faits élémentaires qui étaient nécessaires à la construction de notre base de données. Ces faits élémentaires se retrouvent dans le fichier *"faitselementaires.pdf"*. Ce fichier intègre en même temps une "population" afin de donner des exemples pour chacune des différentes données.

A ce stade du projet, nous avons dû choisir les extensions que nous implémenterons, la description de celles-ci se trouve dans la suite de ce rapport, dans le section "Extensions".

Ensuite, grâce au logiciel *"Dia"*, nous avons créé un schéma conceptuel ORM (*DiagrammeORM_GroupeF.png*) afin de visualiser les liens entre les différentes entités de notre système. Nous avons également dû indiquer sur ce schéma les contraintes d'unicité ainsi que les rôles qui étaient obligatoires.

Par la suite, nous avons traduit ce schéma conceptuel ORM en un schéma relationnel se trouvant dans le fichier "". Cette étape a permis de rendre beaucoup plus simple la création de notre base de donnée.

Enfin, nous avons encore réalisé deux étapes. La première consiste en la création de la base de données proprement dite (fichier Bartender.sqlite) tandis que la seconde est la création d'une liste reprenant toutes les commandes qui nous ont permis de tester si notre base de donnée avait été correctement construite. Cette dernière se trouve dans ce rapport dans la section "Requête SQL".

Extensions

Première extension : "Historique" L'utilisateur de l'application aura l'occasion de consulter un historique, différent selon son statut :

— Le client pourra voir quels achats il a effectués lors du mois écoulé, lui permettant ainsi de savoir à quelle fréquence il se rend dans le bar, qu'est-ce qu'il y consomme, combien il y consomme, lui permettant même ainsi de pouvoir faire attention à sa consommation d'alcool ou, pourquoi pas, à l'inverse (mais cela n'est éthiquement

pas recommandé), de faire des concours avec ses amis pour qui boira le plus.

— Le patron pourra quant à lui consulter un historique de l'ensemble des achats effectués dans son bar, et cela lui servira de base à des statistiques très utiles sur les habitudes de ses clients, lui permettant de prévoir les stocks à avoir, comment adapter les prix, pour quels produits effectuer des promotions, etc.

Deuxième extension : "Préférences" Cette extension va de paire avec l'historique : en effet, le client aura des préférences explicites et implicites.

— Explicites : le client pourra indiquer dans son profil son âge, sa religion et ses allergies, ce qui permettra à l'application de ne pas lui proposer certains produits proscrits.

— Implicites : grâce à l'historique qui permettra à l'application de savoir quels produits sont généralement consommés par le client, celle-ci pourra adapter l'ordre de la carte pour lui proposer en premier lieu les boissons préférées du client.

Requête SQL

— Compter combien de types de boissons différentes on peut trouver sur la carte de boissons.
Select Count(Nom) FROM Boisson

— Trouver toutes les boissons qui font partie d'une commande donnée.
Select Boisson FROM Consommation WHERE AddNum=1

— Calculer le total pour une commande donnée.
Select sum(B.PRIXVENTE*C.Qté) FROM Consommation C, Boisson B, Addition A WHERE A.AddNum=1 AND A.AddNum=C.AddNum AND B.Nom=C.Boisson

— Trouver toutes les boissons, et leur nombre, vendues par un serveur donné.
Select Con.nom, SUM(Con.nboisson) FROM addition Add, consommation Con WHERE Add.login_serveur = "Serveur Donné" AND
Con.IDaddition = Add.IDaddition GROUP BY Con.nom

— Calculer l'addition pour une table donnée, sachant que cette table peut avoir fait plusieurs commandes.
Select sum(C.Qté) FROM Boisson B, Consommation C, Addition A, Utilisateur U WHERE U.Login='AMaalouf' AND U.Login=A.ServeurLogin AND
A.Num=C.AddNum AND C.Boisson=B.NOM GROUP BY B.NOM

— Trouver toutes les boissons dont ils ne restent plus assez en stock (qui sont en dessous du seuil et qui doivent donc être commandées chez le fournisseur).
Select B.Nom FROM Boisson B WHERE B.Stock<B.Seuil

— Trouver toutes les boissons contenant un allergène du Client
Select Ai.Login,Ae.NomBoisson FROM Allergene Ae,Allergies Ai WHERE Ae.Allergene=Ai.Allergene AND Ae.Login=Abra

— Trouver toutes les boissons qui ne contiennent aucun allergène d'un client donné par son login
Select B.nom FROM Boisson B, Allergies_boisson Ab, Allergies A WHERE B.nom = Ab.nom AND A.login = Abra AND A.allergène <> Ab.allergène